Gerenciar permissões de arquivo e propriedade

Sumário

Índice de tabelas

Índice de Figuras

Capítulo 1

Gerenciar permissões de arquivo e propriedade

1.1. Objetivos

- Gerenciar permissões de acesso em arquivos regulares e especiais, bem como diretórios;
- Usar os modos de acesso, como SUID, SGID e sticky bit para manter a segurança;
- Como mudar a máscara de criação de arquivo.

1.2. Mãos a obra

Em sistemas GNU/Linux o ponto forte na administração de usuários, está no sistema de permissão de arquivos. O administrador determina quais usuários e grupos tem acesso de leitura, escrita e execução no sistema.

Como exibir as permissões de um arquivo?

Através do comando **ls** é possível conhecer as permissões de um arquivo, elas são separadas por dono, grupos e outros.

Dono (u) – Permissão ao dono do arquivo, quem por exemplo criou;

Grupo (g) – Permissão ao grupo dono do arquivo, o grupo do usuário que criou;

Outros (o) – Permissão a outros, usuários quem não fazem parte do grupo.

Vamos a prática

 # ls -l /etc/passwd

```
-rw-r--r-- 1 root root 1260 Set 22 04:19 /etc/passwd
```

Em nosso exemplo foi listado com detalhes as propriedades do arquivo /etc/passwd. Vamos detalhar as informações exibidas:

– → Tipo de objeto (arquivo);

rw-r--r-- → Sistema de permissão (U G O);

1 → Numero de objetos no sistema;

root → Dono do arquivo;

root → Grupo dono;

1260 → Tamanho;

Set 22 → Data de criação;

04:19 → Hora de criação;

/etc/passwd → Localização e nome do arquivo.

Em tipo de objeto a letra “d” é exibida quando listamos com detlahes as propriedades de um diretório. Vamos a prática:

Vamos a prática

 # ls -ld /etc

```
drwxr-xr-x 87 root root 6144 Set 27 16:57 /etc
```

Outras letras que representam os tipos de objetos:

l → Link simbólico;

c → Dispositivo de caracter;

b → Dispositivo de bloco;

p → Canal fifo;

s → Socket.

Para pesquisar no sistema por tipo, use o comando find:

 # find / -type l

Em nosso exemplo a opção “type” com “l” foi usada para pesquisar em todo sistema, arquivos do tipo Link simbólico.

Permissão Octal e Literal

A saída do comando ls -l exibe as permissões em forma literal (rwx), mas podemos exibir também de forma octal através do comando stat.

Vamos a prática

 # stat /etc/passwd

```
  File: `/etc/passwd'
  Size: 1260            Blocks: 4          IO Block: 1024   arquivo comum
Device: 301h/769d       Inode: 79116       Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2010-09-27 18:02:27.000000000 -0300
Modify: 2010-09-22 04:19:41.000000000 -0300
Change: 2010-09-22 04:19:41.000000000 -0300
```

Na linha “Access” podemos ver a permissão 0644 em forma octal e -rw-r--r-- em forma literal. Vamos entender os dois modos:

Modo literal

r → Permissão de leitura;

w → Permissão de escrita;

x → Permissão de execução.

A letras aparecem nos três níveis de permissão (U G O)

U	G	O
rwx	rwx	rwx

Cada letra tem um valor que somada define a permissão octal.

U	G	O
421	421	421

O arquivo /etc/passwd tem permissão octal 644

U	G	O
rw-	r--	r--
4+2	4	4

Alterando permissões

É possível alterar as permissões de um arquivo de forma literal e octal, através do comando chmod. Vamos a prática:

1Primeiro crie um arquivo de nome carta.txt

```
$ touch carta.txt
```

Exiba as permissões do arquivo em octal e literal.

$ stat carta.txt

```
  File: `carta.txt'
  Size: 0               Blocks: 0          IO Block: 4096   arquivo comum vazio
Device: 309h/777d       Inode: 79579       Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   aluno)   Gid: ( 1000/   aluno)
Access: 2010-09-27 18:34:12.000000000 -0300
Modify: 2010-09-27 18:34:12.000000000 -0300
Change: 2010-09-27 18:34:12.000000000 -0300
```

Para alterar com o comando chmod:

$ chmod u=rwx,g=rw,o=rw carta.txt

ou

$ chmod 766 carta.txt

Veja as novas permissões

$ stat carta.txt

```
Access: (0766/-rwxrw-rw-)  Uid: ( 1000/   aluno)   Gid: ( 1000/   aluno)
```

Mudar dono e grupo

É possível alterar o dono e/ou o grupo de um arquivo através dos comandos chown e chgrp. Vamos a prática:

Alterando o dono do arquivo carta.txt

```
# chown joao carta.txt
```

Alterando o grupo do arquivo carta.txt

```
# chgrp vendas carta.txt
```

Confira o resultado através do comandos ls

```
# ls -l  /home/aluno/carta.txt
```

Permissão padrão

A permissão padrão para novos arquivos e diretórios, pode ser configurada através do comando umask. Você pode tanto exibir a permissão atual como altera-la.

```
# umask
```

0	**0**	**2**	**2**
Permissão Especial	Usuário	Grupo	Outros

A permissão padrão de um arquivo é de 0666 e de um diretório é de 0777, ao criar um arquivo ou diretório é calculado uma nova permissão, subtraindo o valor do umask da permissão padrão. Exemplo:

0666 – 0022 = 0644 para arquivos

0777 – 0022 = 0755 para diretórios

Regra do umask

Para arquivos existe um regra onde é aplicada quando o valor do umask é par e impar. Exemplo:

0666 – 0015 = 0662 → Subtrai de 7 e não de 6.

0666 – 0022 = 0644 → Subtrai de 6.

Permissão Especial

Além das permissões RWX (leitura, escrita e execução) é possível definir permissões especiais que envolvem segurança ao sistema e privacidade ao usuários. Veja a descrição de cada uma:

SUID → Tipo de permissão especial onde é aplicada em arquivos executáveis, para que eles rodem com as mesmas permissões do seu dono, e não com as permissões do usuário que esta executando. Exemplo:

 # stat /usr/bin/passwd

```
Access: (4755/-rwsr-xr-x)  Uid: (    0/    root)   Gid: (    0/    root)
```

Um arquivo com permissão SUID recebe o numero “4” e um “s” no lugar do “x” no campo de execução para dono.

SGID → Tipo de permissão especial onde é aplicada em arquivos executáveis e em diretórios, para que eles rodem com as mesmas permissões do grupo dono, usado para limitar o acesso a um executável do sistema a um determinado grupo. Exemplo:

 # stat /usr/bin/chage

```
Access: (2755/-rwxr-sr-x)  Uid: (    0/    root)   Gid: (   42/  shadow)
```

Um arquivo com permissão SGID recebe o numero “2” e um “s” no lugar do “x” no campo de execução para grupo.

STICKY → Tipo de permissão especial onde é aplicada em diretórios, que impede que usuários apaguem arquivos não criados por eles mesmos. Exemplo:

```
# stat /tmp
```

```
Access: (1777/drwxrwxrwt)  Uid: (    0/    root)   Gid: (    0/    root)
```

Um diretório com permissão STICKY recebe o numero “1” e um “t” no lugar do “x” no campo de execução para outros.

Capítulo 2

Gerenciando

2.1. Objetivos

- Trobleshooting: Gerenciar permissões de grupos para conceder acesso a arquivos e permissões especiais.

2.2 Troubleshooting

Como configurar permissões especiais para grupos?

Vamos criar um cenário onde um diretório publico na empresa pode ser acessado por todos os usuários, mas os arquivos criados nele devem pertencer a o grupo do diretório e não ao grupo do usuário que criou o arquivo. Vamos a prática:

Primeiro vamos criar o diretório publico

 # mkdir /publico

Crie também o grupo que será o grupo dono.

 # addgroup vendas

Altere o grupo do diretório para vendas

 # chgrp vendas /publico

Adicione a permissão especial SGID ao diretório publico

 # chmod 2777 /publico

Verifique as permissões do diretório

 # stat /publico

```
Access: (2777/drwxrwsrwx)  Uid: (    0/    root)   Gid: ( 1002/  vendas)
```

Acesse o diretório publico com um outro usuário e crie um novo arquivo

$ cd /publico & touch teste.txt

Verifique o dono e grupo do arquivo criado

$ ls -l teste.txt

```
-rw-r--r-- 1 aluno vendas 0 Set 27 21:52 teste.txt
```